Synthesizing Meaningful Feedback for Exploring Virtual Worlds Using a Screen Reader

Bugra Oktay
Department of Computer Science and Engineering
University of Nevada, Reno
Reno, Nevada, USA
oktayb@unr.nevada.edu

Eelke Folmer
Department of Computer Science and Engineering
University of Nevada, Reno
Reno, Nevada, USA
efolmer@unr.edu

Copyright is held by the author/owner(s).
CHI 2010, April 10–15, 2010, Atlanta, Georgia, USA.
ACM 978-1-60558-930-5/10/04.

Abstract
Users who are visually impaired can access virtual worlds, such as Second Life, with a screen reader by extracting a meaningful textual representation of the environment their avatar is in. Since virtual worlds are densely populated with large amounts of user-generated content, users must iteratively query their environment as to not to be overwhelmed with audio feedback. On the other hand, iteratively interacting with virtual worlds is inherently slower. This paper describes our current work on developing a mechanism that can synthesize a more usable and efficient form of feedback using a taxonomy of virtual world objects.

Keywords
Virtual Worlds, Accessibility, Audio I/O

ACM Classification Keywords
H.5.2 [User Interfaces]: Voice I/O

General Terms
Human Factors, Measurement, Design

Introduction

Virtual worlds have enjoyed increasing popularity during past years, with millions of participating users. The immersive graphics, the large amount of user-generated content and the social interaction opportunities offered by the greater sophistication of virtual worlds, could eventually make for a more interactive and informative World Wide Web. Popular virtual worlds include Second Life [4] and World of Warcraft [1]. The focus of our research is on virtual worlds with user generated content (which have no elements of combat) as these are increasingly used as cyber learning environments [7].

In Second Life, users control a digital puppet, called an avatar–with human capabilities, such as walking and gesturing–through a game-like, third person interaction mechanism. Until recently [2,3,6,9] virtual worlds were inaccessible to users who are visually impaired as these virtual worlds are entirely visual and lack of any textual representation that can be read with a screen reader or tactile display.

In our previous research, we developed a screen reader accessible interface for virtual worlds called TextSL [2,8]. TextSL allows screen reader users to access Second Life and interact with large numbers of objects and avatars there, using a command-based interface that is inspired by multi-user dungeon games. Users navigate their avatar using commands such as: *"move"* or *"teleport"*. Users can query their environment using the *"describe"* command, which lists the number of objects and avatars found within a 360 degree 10 meter radius around the user's avatar. Objects and avatars can then be iteratively queried (See Figure 1).

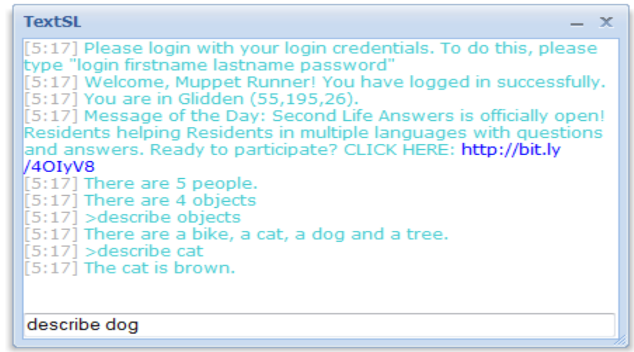

Figure 1. It takes two *"describe"* commands to learn that the cat is brown.

User generated virtual worlds, are densely populated with objects, e.g., in Second Life we found that on average 13 objects can be found within a 10 meter radius around the user's avatar [2]. Providing all the object names as audio feedback may easily overwhelm the user, especially if the names of the objects are long, which motivated the use of a mechanism where users have to iteratively query their environment.

User studies with TextSL show that a command-based interface is feasible [2], as TextSL allows screen reader users to explore Second Life, communicate with other avatars, and interact with objects with the same success rates as sighted users using the Second Life viewer (TextSL has been designed to support access to other open source virtual worlds such as OpenSim [5] once APIs become available). However, command-based exploration and object interaction is significantly slower in TextSL [2] due to the requirement of users to have to iteratively query their environment. Some users found the amount of feedback that TextSL provides overwhelming. The focus of our current research is to synthesize a more meaningful form of feedback that seeks to balance: (1) minimizing the number of overwhelming feedbacks; and (2) minimizing the amount of interaction required.

Synthesizer

Users who are visually impaired typically use their screen readers at different speech rates, which indicates that screen reader users have different abilities to process audio feedback. The proposed synthesizer incorporates a user specified word limit (UWL). Since words may vary in length and this will take different amounts of time to be pronounced through a screen reader, in future work, the UWL could be combined with a user specified time limit. However, this would require TextSL to know the speech rate of the screen reader.

The synthesizer executes as follows:

1. Scan and filter objects within a fixed range around the user and compile the found names into the Scanned Word List (SWL)
IF (#SWL > UWL)
2. Group and aggregate SWL.
ELSE
3. Detail SWL.

Step 2 specifically focuses on compressing the description to prevent overwhelming the user with feedback and step 3 focuses on reducing the number of *"describe"* commands that must be given. The synthesizer will either execute step 2 or 3 depending on the number of words generated from step 1. Although nearby avatars are parts of the provided feedback and they are just as important as the objects, the synthesizer only focuses on virtual world objects because (1) any avatar can be of importance to the user regardless of its properties so filtering is not applicable; (2) the number of objects around the user is typically much larger than the number of avatars and therefore feedback is most effectively synthesized through grouping and aggregating objects.

Object Scanning and Filtering

The Second Life client displays objects and avatars that are in front of the user's avatar. To eliminate the user's need of turning to different directions to find out what is there, TextSL considers all objects within a 360-degree radius with a user specified range (default is 10 meters) around the user (Figure 2).

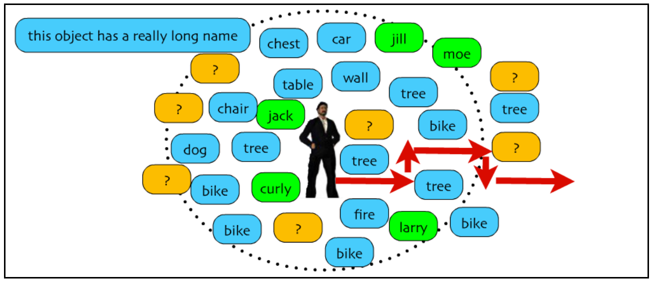

Figure 2. Scanning of objects around the avatar.

For each object we compute a value according to the following function:

$$F = \text{NAME} * \text{SIZE} * \text{DISTANCE}^{-1} * \text{INTERACTION} * \text{ROOT}$$

Where,
NAME: The length of the name of the object divided by the average word length. Objects with non-descriptive names like *"object"* are given the value 0.

SIZE: The bounding box of the object.
DISTANCE: Distance in meters to the user divided by the scanning range.
INTERACTION: 10 if the object allows interaction and 1 if not. ROOT: 1 if the object is the root object and 0 if the object is a sub object.

This function prioritizes objects that: (1) have more descriptive names; (2) are closer to the user; (3) are larger; (4) are interactive; and (5) are not sub objects. The latter is to avoid users interacting with parts of a larger object such as a wheel, which is part of a larger object such as a car. As most content creators assume that users can see, they frequently leave the name of objects to their default value ("object"). This is a problem when users query their environment as

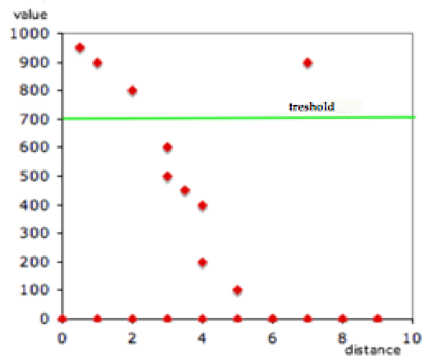

Figure 3. Sample output value function for a number of objects within 10 meters of the user.

this may return the names of multiple objects called "object", which are basically meaningless to TextSL users. We identified that for Second Life 32% of the objects are called "object" [2]. Such objects are culled from our object scanning. Only objects with a value above a certain user specified threshold value are compiled into the SWL (See Figure 3).

Grouping and Aggregation

If the number of words in the SWL is over the UWL limit, then we need to reduce this by grouping and aggregation.

Grouping: Objects with the same name are grouped together, e.g., *[car, car, dog] → [2 cars, dog].* Grouping has no information loss but this step may not significantly reduce the number of words if the number of grouped objects is below three. Some savings are incurred when adjectives are included in the UWL count but as these typically are very short we choose not to include these. Still, saying "There are 2 cars." makes more sense than saying "There are a car and a car."

Aggregation: Object names are aggregated if they can be determined to be members of the same class. Aggregation requires the use of a taxonomy of virtual world objects. This taxonomy is something that we are currently creating in related work where we developed a game within Second Life that can help improve the accessibility of virtual worlds as often meta-data for virtual world objects is missing. In this game sighted users can tag and label objects using a scavenger hunt game, which builds a set of training examples for an automatic classifier that can recognize objects not having a name based on their geometry. This game also helps build taxonomy of virtual world objects, which we can use in aggregating a more usable form of feedback. The taxonomy we create is described as a set of rules, e.g., *[vehicle→car]* or [*animal→dog]* and these rules may also define subtypes *[dog→poodle].* The taxonomy of objects created as such is not restricted to

Second Life and can be used by any virtual world that has textual descriptions for their objects.

Using the taxonomy, we analyze whether any of the object names can be aggregated to the same parent, *e.g., [bicycle, car] → [2 vehicles].* Larger reduction in word count can be achieved when objects can be aggregated to the highest possible class, *e.g. [cat, dog, bird] → [3 animals].* However, this may also yield a much higher level of information loss, e.g., *[poodle, mastiff] → [2 animals]* and *[poodle, mastiff] → [2 dogs]* are both valid aggregations. The first example has significantly higher information loss that would still require the user to iteratively query the object set, which is what we are trying to avoid. The second transformation may require a more detailed taxonomy of virtual world objects that also includes subtypes, which may be more costly to create. Aggregation transformations are only applied if they reduce the number of words in SWL. Figure 4 shows example output of the grouping and aggregation steps.

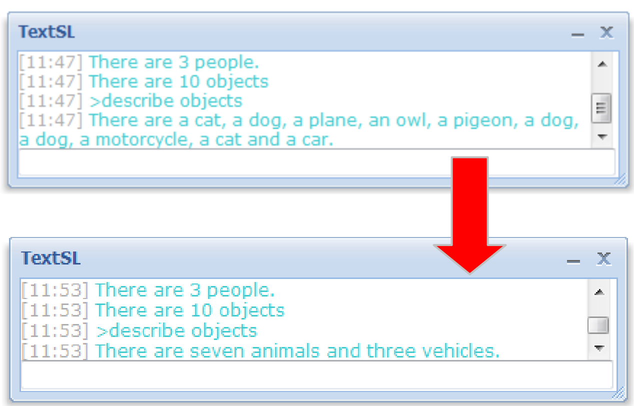

Figure 4. Sample output after Grouping and Aggregating.

Detailing

If the number of words in the SWL is below UWL then in order to reduce the amount of interaction required, we detail the objects in the SWL with specific object information such as color or size. For example: *[cat] → [big red cat].* Specific transformations are only applied as long as we do not exceed the UWL with a 10% margin.

In addition to the *"describe"* command users can issue a *"where"* command indicating where the object is relative to the user's avatar. Spatial information can also be added to objects during detailing to further reduce interaction e.g., *[cat] → [cat in front of you].* We consider four spatial locations (left, right, behind, in front). Adding spatial information requires grouping of objects based on location to reduce the number of words in the SWL, for example, *[cat to your left, car to your left] → [a cat and a car to your left]*.

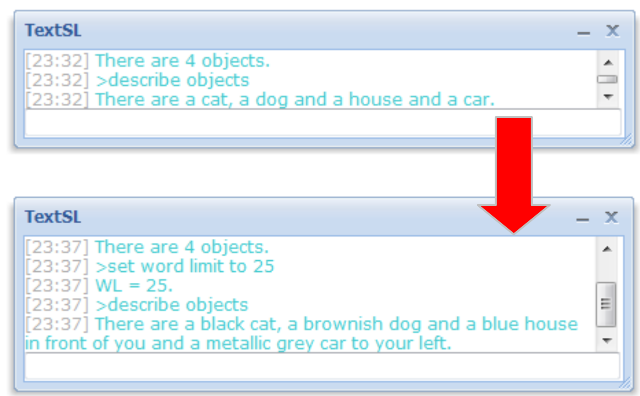

Figure 5. Sample output with detailing implemented.

Users can assign priority values to properties (location, size, color) with respect to their importance. Specific transformations to detail objects are only applied when it can be applied to all the objects in the SWL in order to ensure consistency of feedback. Figure 5 shows an example output of TextSL with detailing implemented.

Future Work

Currently, only a limited number of taxonomy rules have been defined manually and these describe a simple taxonomy for virtual world animals and vehicles which allowed for implementing the proposed synthesizer in TextSL. We seek to collect more labeling efforts through our scavenger hunt game that will allow for expanding our current taxonomy. Once this has been established, we will evaluate the effectiveness and usability of synthesizing more usable forms of feedback through a series of user studies, where different forms of synthesizing feedback will be explored.

Conclusion

The large amount of objects in virtual worlds poses a significant problem for text-based approaches towards making virtual worlds accessible to users who are visually impaired. The amount of feedback provided may overwhelm the user and consequently iteratively querying a user's virtual surroundings is slow. We seek to provide more usable forms of information by transforming the feedbacks about a user's virtual environment into more concise or descriptive forms using a taxonomy of virtual world objects.

Acknowledgements

This work is supported by NSF Grant IIS-0738921.

References

[1] Blizzard Studios, World of Warcraft, http://www.worldofwarcraft.com.

[2] Folmer, E., Yuan, B., Carr, D., Sapre, M. Textsl: A Command-Based Virtual World Interface for the Visually Impaired. Proc. ACM ASSETS '09, pp 59-66 2009.

[3] IBM Human Ability and Accessibility Center, Virtual Worlds User Interface for the Blind, http://services.alphaworks.ibm.com/virtualworlds/

[4] Linden Research, Second Life, http://www.secondlife.com/

[5] OpenSim, http://opensimulator.org/

[6] Pascale, M., Mulatto, S., Prattichizzo, D. Bringing haptics to Second Life for visually impaired people. Proc. EuroHaptics 2008, pp 896–905, 2008.

[7] Robbins, S.S. Immersion and engagement in a virtual classroom: Using second life for higher education. In EDUCAUSE Learning Initiative Spring 2007 Focus Session, 2007.

[8] TextSL, Screenreader accessible interface for Second Life, http://textsl.org

[9] Trewin, S., Hanson, V., Laff, M., Cavender, A. Powerup: An accessible virtual world. Proc. ACM ASSETS '08, pp 171–178, 2008.